energies

MDPI

Article

A New Approach for Real Time Train Energy Efficiency Optimization

Agostinho Rocha [1,*], Armando Araújo [1], Adriano Carvalho [1] and João Sepulveda [2]

[1] Department of Electrical and Computer Engineering, Engineering Faculty, University of Porto, 4200-465 Porto, Portugal; asa@fe.up.pt (A.A.); asc@fe.up.pt (A.C.)

[2] Department of Industrial Electronics, University of Minho, 4800-058 Braga, Portugal; mjs@dei.uminho.pt

* Correspondence: up200706597@fe.up.pt

Received: 2 August 2018; Accepted: 26 September 2018; Published: 5 October 2018

Abstract: Efficient use of energy is currently a very important issue. As conventional energy resources are limited, improving energy efficiency is, nowadays, present in any government policy. Railway systems consume a huge amount of energy, during normal operation, some routes working near maximum energy capacity. Therefore, maximizing energy efficiency in railway systems has, recently, received attention from railway operators, leading to research for new solutions that are able to reduce energy consumption without timetable constraints. In line with these goals, this paper proposes a Simulated Annealing optimization algorithm that minimizes train traction energy, constrained to existing timetable. For computational effort minimization, re-annealing is not used, the maximum number of iterations is one hundred, and generation of cruising and braking velocities is carefully made. A Matlab implementation of the Simulated Annealing optimization algorithm determines the best solution for the optimal speed profile between stations. It uses a dynamic model of the train for energy consumption calculations. Searching for optimal speed profile, as well as scheduling constraints, also uses line shape and velocity limits. As results are obtained in seconds, this new algorithm can be used as a real-time driver advisory system for energy saving and railway capacity increase. For now, a standalone version, with line data previously loaded, was developed. Comparison between algorithm results and real data, acquired in a railway line, proves its success. An implementation of the developed work as a connected driver advisory system, enabling scheduling and speed constraint updates in real time, is currently under development.

Keywords: railway; train; energy efficiency; driver advisory system; optimization algorithms; simulated annealing; scheduling

1. Introduction

Nowadays, energy efficiency has become a permanent issue as the planet's resources are limited and the demand for energy increases continuously due to either an increasing population or access of an increasing population to standards of living demanding more energy.

An increasing mix of renewable and fossil energy sources to satisfy the needs of energy is observable.

Energy use is becoming evermore controlled by automation of industrial processes, buildings and transportation systems. Simultaneously, the main world economies have shown that it is possible and necessary to reduce the consumption of energy without prejudicing the quality of life, by addressing the issue of energy efficiency to within sectors of human activity.

Railways are one of the sectors demanding intensive energy and power. Therefore, managing energy consumption is a significant issue for society. Therefore, energy efficiency in railways is a popular research topic leading to the development of new technologies and algorithms. Due to

operational requirements these technologies are used as tools assisting train drivers and are usually called Driver Advisory System (DAS).

In fact, DAS use in railway lines can reduce energy consumption, increase railway capacity and reduce driver training time. DAS are installed in the driver cabin and usually determines Optimal Speed Profile (OSP)s dictated by scheduling, velocity limits, line shape and spent energy. Also, they are used to advise the train driver about the best driving mode for the actual journey. OSPs generated by the system define the operating velocities for cruising, coasting and braking along the journey. Those variables are highly important due to their energy consumption dependency [1].

First developments on DAS started in the 70s, as mentioned in [2], with optimal control theory. In his PhD thesis, Milroy uses Pontryagyn's maximum principle, applied to each train pathway section, with the purpose of finding the OSP which allowed minimum energy consumption [3–5]. One of the most important outcomes of Pontryagyn's maximum principle, applied to this energy minimization problem, was the shape definition and driving regimes sequence to be applied to a train's movement between two consecutive stations, which minimizes energy consumption. Four driving regimes were identified based on traction and braking force values [2,6]:

- Acceleration—Driving regime applied, typically, at the beginning of the journey. It applies maximum force to move the train from starting up to the so-called cruising velocity. This regime uses the maximum available force to maximize acceleration. When at cruising velocity, this regime ends.
- Cruising—After reaching cruising velocity, next driving regime is characterized by constant force, so zero acceleration, and, consequently, constant velocity. This regime time interval is a variable to be determined by problem objectives (usually energy minimization).
- Coasting—To reduce energy consumption, a coast phase is added to optimal velocity profile. During this regime, no traction force is applied and, consequently, energy consumption is zero; the end of this period is also a variable to be determined (also constrained by problem objectives). In fact, good choices for coasting points shortens traction regimes. However, it tends to increase journey time, which is constrained to schedule. So, the points in the journey where coasting should start and finish are very important ones in terms of minimizing energy consumption [7–9].
- Full braking—Driving regime applied at the end of a journey to guarantee the train stops at the scheduled time and correct position. This regime uses of the maximum available braking force to maximize deceleration.

Figure 1 shows the four driving regimes, applied between two stations, associated with the OSP. This profile assumes a line with zero slope and no velocity constraints. As shown, the train's movement starts with a phase of maximum acceleration followed by a cruising phase. The acceleration phase ends when the train reaches the pre-defined cruising velocity. Next a coasting phase starts with zero traction force. This traction regime reduces energy consumption. As expected, minimum energy consumption is reached by longer coasting regimes but, unfortunately, travelling time increases. Therefore, a period of coasting phase is determined as a trade-off between energy consumption and total travelling time. The last phase is a braking one that ensures station arrival at zero speed at the schedule time. Although the optimum profile uses a flat line, its application in the presence of small gradients is still possible. Nevertheless, searching for OSPs for real situations must use line slopes and velocity limits. In fact, the presence of slopes and velocity constraints in the line will cause a break in each traction regime. As an example, the existence of a maximum velocity limit can interrupt a cruising phase and introduce a braking one. A new acceleration phase and another cruising one will then follow. Other changes to OSPs can happen in situations where, for instance, there is no space to apply all driving regimes, or a different sequence must be applied. This usually happens in suburban or metro services [2,10,11].

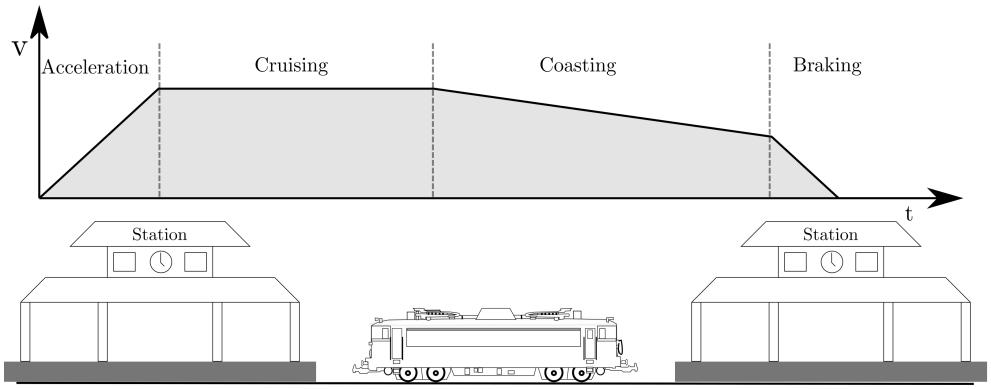


Figure 1. Driving regimes associated to the OSP.

Therefore, using Pontryagyn's Maximum Principle (PMP) for the solution to the general problem, associated with the identification of the points related to the four identified driving regimes, and solving for the resulting optimization problem, is a hard task and not usually compatible with real-time use.

Recently, some research in the field of meta-heuristics, using nature-inspired structures and solution-searching mechanisms, has emerged as an alternative to the use of PMP for determination of OSPs. These are the cases of Evolutionary Algorithm (EA) [7–9,12,13], Ants Colony Optimization (ACO) [14,15], Particle Swarm Optimization (PSO) [16] and Simulated Annealing (SA) [17,18].

The use of meta-heuristics for real-time DAS application gives more flexibility to problem formulation, enabling the easy updating of line data and timetable to actual operating conditions. In addition, algorithm running time has to be compatible with its application to the connected DAS. These are very important characteristics, associated with this kind of approach, in the context of its application to DAS.

Despite its proved efficiency, only a few works deal with the SA algorithm for the problem of energy-efficient train control. The algorithm copes with non-linear models subjected to many constraints, it is flexible and versatile, it statistically converges to an optimal solution, and it is fast-tuned [19]. Therefore, this paper presents a new algorithm for OSPs generation, with a solution-searching mechanism based on SA and focused on reducing the amount of consumed energy in train operation. The strategic approach, for OSP solution, applies a SA algorithm constrained not only to scheduling, speed limits and line profile, but also to driving comfort, imposing limits on maximum acceleration and enabling the use of the four regimes defined by application of the theory of optimal control. A good selection of initial annealing temperature, allied to the mechanism used for the generation of new cruising and braking velocities, enables the accomplishing of real-time requirements, namely a short computation time. A comparison between these new algorithm results and actual train data, in four different routes, proves that it can find the OSP for minimum energy in real time.

1.1. Paper Outline

The paper is organized as follows: Section 2 introduces some of the work already done concerning the use of meta-heuristics for solving the proposed problem, as well as its most common limitations. The main results and contributions are also presented. Section 3, Train Dynamics, presents the model adopted for the train and its associated motion simulator, Train Motion Simulator (TMS) followed by some discussion about OSP generation. In the next section, Speed Profile Optimization analyzes the problem of finding, between all generated speed profiles, the one that minimizes consumed energy, E_c. As the problem has several constraints, related to passenger comfort, line gradients, travelling times and speed limits, the solution is tried using the SA algorithm, which is presented alongside the used optimization procedure. Section 5, Results, shows results obtained by simulation and compares it with real logged train data. Finally, Section 6 presents main conclusions and work to be done.

1.2. Notation

In this work traction force is denoted by F_T and its maximum by $F_{T_{max}}$. Braking ones are F_B with a maximum of $F_{B_{max}}$. Resistive and gravitational forces are, respectively, F_R and F_G. For train position, velocity and acceleration, at times t and $t-1$ it is used, respectively, x_t, x_{t-1}, v_t, v_{t-1} and a_t, a_{t-1}. Train equivalent mass uses M_e, being γ the mass correction factor. Mechanical power is P_{mec}. E_c the energy associated with a journey between two stations. In what concerns velocities, related to start of cruising and start of braking, they are, respectively, v_{op} and v_{bk}. Its values at iteration i are v_{op}^i and v_{bk}^i and its variations are denoted by Δv_{op}^i and Δv_{bk}^i. Accelerations during traction, cruising, coasting and braking are a_{acc}, a_{cru}, a_{coa} and a_{bk}. Distances made during those regimes are, respectively, Δ_{xacc}, Δ_{xcru}, Δ_{xcoa} and Δ_{xbk}. Constraints associated with passenger comfort, scheduling, and line are maximum acceleration, a_{max}, journey time, t_{max} and speed limits, v_{max} and v_{min}. The objective function and temperature used in SA algorithm, at iteration i, are denoted by O_i and T_i. SA penalty factor, cooling rate and maximum iterations are $lambda$, s and it_{max}.

2. Literature Review

Search algorithms have been recently used for the OSP problem when applied to real train pathways constrained to its velocity limits, gradients and timetables, with the purpose of energy consumption minimization. The most common applied meta-heuristics uses Artificial Neural Network (ANN), ACO, Dynamic Programming (DP), Genetic Algorithm (GA), PSO or SA.

In [7] the authors present a GA for coast point determination in a train pathway. It generates a new solution at each train stop, and before departure, producing a coast control table. The use of a coast control table is justified by the possibility to use it in an Automatic Train Operation (ATO) system. It searches for the optimum solution considering a multi-objective function. Each coast profile result is constrained to time punctuality, passenger comfort and energy consumption. The algorithm results for two different cases, based on schedules definition, were compared with a fuzzy logic control ATO.

The algorithm does not have a cruising phase, using several cycles of motoring and coasting, making the trip uncomfortable. Also, nothing is said about the used TMS and the possible line constraints. The algorithm is implemented offline with a pre-determined number of coasting points. This is so because simple GA do not have the capability to make this selection. Each point has an associated gene, with 10 bits, for distance discretization. So computation time is higher for longer trips if spatial resolution is to be maintained. Nevertheless, a C++ implementation of the algorithm typically runs in half a minute in an IBM 486-compatible PC. Therefore, the authors claim that it has potential for online implementation. The gains in energy, in the two reported cases, are small and about 7% and 3%, respectively.

Another GA is presented in [8]. It also locates coast points in a train pathway. At each stop, a new profile for the travelling distance between two consecutive stations is determined. It uses a binary string, with a variable length, dependent on the total travelling distance, to represent the coast point's position. The use of a binary string intends to reduce the complexity of mutation and crossover operations used during the optimization process. The authors also considered, beyond a single station-to-station optimization, a multiple-station scenario. For this last scenario, a Hierarchical Genetic Algorithm (HGA) was implemented. It uses previous string structure with an extra variable representing total coast points needed. The algorithm uses a Minimum-Allele-Reserve-Keeper (MARK) mutation scheme, in order to reduce processing time. Solution results use a cost function based on two parameters: schedule time and energy consumption.

The algorithm uses only one or two cycles of motoring and coasting and does not have the cruising phase. A simple GA lacks the capability to select the number of coasting points, an HGA is tried, but only for two coasting points, so does not guarantee a near optimal solution. Also, in order to meet the request for real time, a simpler mutation scheme is used. TMS and associated constraints are not mentioned. Presented energy gains are 30% but at the expense of 30% extra trip time, which is not always possible. The algorithm was implemented in Visual Basic to achieve a good human

interface. However, Visual Basic is a proprietary programming language not easily transferred to other operating systems.

In [9], the search for an optimum speed profile, with energy consumption minimization, uses a GA together with an ANN. The purpose of the ANN is to substitute the train dynamic model. ANN inputs are a sequence of coast points and its outputs are total time and energy consumption. GA determines the best option of coast point sequence based on a cost function defined as a weighted sum of travelling time with energy consumption. Algorithm tests on a Turkish metro line with five stations and two lines, on a multi-train situation, proved its effectiveness.

Again, the cruising phase is omitted. Train speed profiles are made with the use of proprietary software, namely SimuX. Being aware of the problems related to GA real-time implementation, namely the huge number of times that the TMS must be called (for the considered population size and number of generations 10,000 times), the authors use as its replacement an ANN without slope constraints. Nevertheless ANN has limitations as it is trained for a specific line with specific constraints and therefore is not flexible enough, e.g., online timetables or line speed limits change. Also, ANN training demands hundreds of offline simulations (made by SimuX) which are extremely time-consuming. Moreover, a search for optimal coasting points is made with Matlab GATool, making real-time implementation very difficult. The two simulated results show, respectively, a 30.85% energy saving, with a 4.81% increase in travel time, and 18.25% less energy associated with a 4.65% increment in travel time.

In [12] the authors present another example with GA: speed profile determination based on a multi-population GA. The speed profile determination uses two phases. The first one uses travelling distance, between two consecutive stations, in order to find the most economical scenario. The second one considers the full trip. Searching process makes use of a multi-population scheme. This enables time travel reduction and avoids the algorithm getting stuck in local minima. Real data from a subway line section in Beijing with a total distance of 21 km was used to test the algorithm. The tests considered line gradients, curve radios and velocity limits.

The aim of the work is minimization of total energy consumption between multiple stations. Inter-station trip time may vary but total time should be relatively constant. This is done again by finding the positions for switching between acceleration, coasting and braking. As usual in GA the cruising phase is missing. As general GA suffers from premature convergence, when associated with these kinds of problems, a multi-population GA is proposed. This has the problem of generating a big number of calls to the used TMS (it is expected 20,000 calls for the 200 generations and 100 individuals proposed in the paper). Therefore, this will make its real-time operation hard. Additionally the method assumes that times between consecutive stations can vary as much as 16% and, usually, this is not the case. Nevertheless, results for the six simulated inter-stations show that the algorithm enables 6.16% energy reduction, maintaining total trip time.

EA are another example of algorithms applied in railways. In [13,20] the authors present a multi-objective one for velocity profile determination. The proposed algorithm is of Indicator—Based Evolutionary Algorithm (IBEA) type. It focuses on minimizing energy consumption and travelling time, by one hand, and energy consumption minimization, total travelling time and delays reduction, by other hand. It divides train pathway, limited by the distance between two consecutive stations, into several sections defined by velocity limits. For each section, it is defined a group of values for velocities profile solution (algorithm uses input and output velocities, as well as, the maximum and desired ones. It also uses velocities limits of next section). Two different lines in France served for algorithm validation.

Again, the main objective is associated with minimizing energy and travel duration. It uses a complex TMS dividing the travel into sections according to line speed profiles. For each one five variables have to be determined.The solution encoding needs a vector with triple the number of sections. The algorithm needs the ParadisEO framework and uses as termination criteria simulation time (60 s). Therefore, its real-time implementation outside the referred framework will be a challenge.

However, simulation results show that energy reductions up to 54% can be achieved with 15% of increase in trip time.

A comparison between three methods is presented in [14]. The proposed problem is the search for optimum speed values along the journey. Points with zero velocity, and with speed limits changes, are the ones where there is a need for velocity calculation. The studied algorithms, for performance comparison, were GA, ACO and DP. These algorithms were used in three situations with different trip times.

Main conclusions are that in both situations DP showed a better performance at the expense of an enormous computing complexity, so it cannot be used in real-time applications. Travel smoothness is poor for both the ACO and the GA algorithms mainly for longer travelling times. Also, in some of the studied cases both the ACO and the GA algorithms were unable to find a solution.

Another example that uses ACO algorithm for speed profile optimization is presented in [15]. The generated speed profiles use a cycle of acceleration, cruising, coasting and braking. Solution is searched by means of a MAX-MIN Ant System (MMAS) in two optimization stages (the first stage is made offline and its results are the reference for second-stage optimization). For its assessment, the algorithm used real train data acquired from a subway line in Beijing.

Algorithm simulation results show a total computing time of about one minute, being 40 s for the first stage optimization and the remaining 20 s for the second one. Energy saving rate is 14%. Therefore, for dwell times bigger than the second stage optimization time, the algorithm can be used in real-time applications.

The application of other nature-inspired searching algorithms, besides GA and ACO, was also recently investigated. In [16] a PSO-based algorithm, with a multi-objective function, was implemented in order to find the Pareto front for energy and trip time. It uses a train equipped with an ATO. The objective is minimization of both energy consumption and running time. The algorithm generates a population of random command sequences and searches for a set of solutions making a Pareto front of minimum energy and possible running times. Its parameter tuning and validation used tests made on a flat track, in combination with a simple speed limit profile, in order to reduce computation time. Algorithm validation uses a line section of the Madrid subway.

Considering the 20,860 possible command combinations for speed profile generation, it needs 50 min of computation using a TMS previously developed by the authors. Obtained Pareto front with the multi-objective PSO-based algorithm has a computation time of about ten minutes. Therefore, it cannot be used for real-time applications. Also, the algorithm searches for pairs of energy/trip time so cannot be constrained to existing timetables. However, in some cases, it can have energy savings of 20%.

In [17] a SA-based algorithm is presented. It intends to reduce energy consumption in the metro line between New York and Connecticut. The algorithm considers line gradients and velocity limits during the searching process. Energy consumption calculation, for each generated solution, uses a dynamic model of the train previously developed by the authors. The TMS considers four motion regimes, accelerating, cruising, coasting, and braking. However, cruising is only applied if needed. The SA algorithm uses a cost function based on minimization of energy consumption and schedule travelling times. Also, it enables re-annealing and uses Metropolis criterion as the rule for solution acceptance. The initial value of temperature is 100 and temperature updates uses an exponential schedule. Results of the algorithm are, for each travelling regime, maximum velocity, and time and position of each coast point.

SA-based algorithms need to define a good starting temperature but nothing is said about that in the paper. Furthermore, the initial solution and the method for generation of new solutions are not presented. A cooling schedule factor is also missing. These are very important factors required for good algorithm output. Regarding results, nothing is said about computation time and obtained energy savings.

In [21] the authors present an algorithm that is a combination between GA and SA, a so-called Genetic Simulated Annealing Algorithm. The objective of joining both algorithms is to eliminate its individual weaknesses and to enhance its advantages. So, in a first stage GA is used to create a good initial value solution for SA. In a second stage SA is initiated. This is to avoid local minima that could occur if GA was only used. Matlab is used for running the TMS. It outputs speed, position and time vectors, and energy consumption of the obtained speed profile. Some stations of the Eskisehir light rail served for algorithm performance tests.

As already mentioned, GA and SA algorithms need well-chosen parameter sets. So, for satisfactory results, some method has to be used for parameter tuning. Therefore, the authors use for this purpose a single 2000 m straight track with gradients, and made six test runs varying GA crossover and mutation rates, as well as selection and crossover function. SA algorithm annealing and temperature functions were also varied. Nevertheless, nothing is said about the criteria used for these variations. The paper concludes that GSA can give very good solutions but it has convergence problems so the tuning of its parameters is crucial. Unfortunately, once again, nothing is said about the gains in energy and total computational time.

Contributions and Main Results

Regardless of being scarcely used for the train energy minimization problem, and, to our knowledge, only being used for offline solutions, the SA algorithm has showed its effectiveness in solving minimization problems. The most common limitations of current works can be overcome by the use of the SA algorithm. In fact SA can solve problems with multi-constraints (comfort, speed limits, gradients, scheduling), can use of the optimal driving strategies for a train (acceleration, cruising, coasting and braking), is of easy implementation in a non-proprietary environment, has a reduced number of parameters, without need for any tuning, it escapes from local minima, and can converge to an optimal solution in a small number of iterations. Also, fast calculation time enables its online use. Therefore, this paper presents a new SA-based algorithm focused on reducing the amount of consumed energy, constrained to comfort, velocity limits, line profile, and timetable scheduling. A good criterion for initial temperature choice [22] and the method adopted for generation of new v_{op} and v_{bk} velocities, for OSP solutions, is presented in Section 4.2, allows real-time operation and avoids it becoming trapped in local optima. Also, the algorithm is implemented in Matlab, enabling a fast migration to C type languages for execution in a DAS. Main results show that this new algorithm can find the OSP for minimum energy consumption in a running time compatible with online operation. Simulations, for four inter-stations, shows maximum computation time of 20 s and average energy savings of 10%.

3. Train Dynamics

As previously stated it is intended to find the OSP associated with the minimization of train energy. Therefore, a dynamic model capable of train time, distance and respective consumed energy calculations, for each of the speed profiles to be analyzed by the SA algorithm, will be needed. This paper uses the mass-point model, presented in [23], that assumes motion of a train, with distributed mass, as the motion of a point in the train centre of mass. Besides, this is an approach commonly used by several authors considering that travelling distances are much larger than train length [24,25]. Following subsections will show the train and motion simulator models needed for OSP generation.

3.1. Train Model

As already indicated, modelling train motion is essential in order to output possible solutions for SA algorithm to work with. So, consider Figure 2 showing all the forces that are acting in the train during its operation.

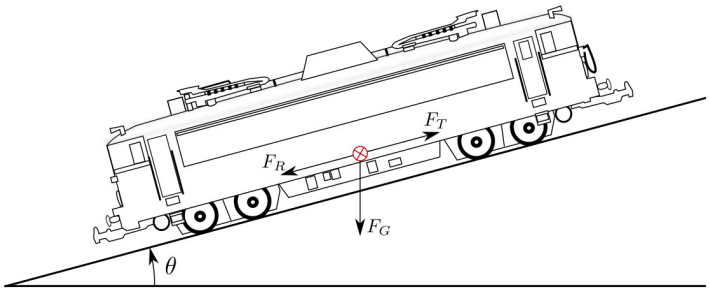


Figure 2. Train acting forces in uphill motion.

Assuming an uphill motion, F_R and F_G represents, respectively, tractive effort resistance and gravitational force associated with mass of the train. These are forces opposed to movement direction. To overcome those forces a tractive effort force must be applied, represented by F_T. This force is positive (F_T) when in acceleration regime and negative (F_B) when in braking one. Equation (1) gives the sum of total applied forces in the train:

$$\sum F = F_T - (F_B + F_R + F_G) \tag{1}$$

Gravitational force, F_G, caused by line slopes is determined by Equation (2):

$$F_G = Mgsin(\theta) \tag{2}$$

where M is the mass of the train, g is the gravitational acceleration and θ the slope angle.

The tractive effort resistance, F_R, presented in Figure 3b is usually represented using Davis equation [26]:

$$F_R = A + Bv + Cv^2 \tag{3}$$

Davis Equation (3) describes train resistance behaviour by making use of a polynomial function. A, B and C parameters have different physical meaning and are determined in function of the vehicle. Parameters A and B are related to mechanical resistance and depend on the train mass. Parameter C is associated with aerodynamic resistance [27]. As v is train velocity, motion resistance force, F_R, is velocity dependent.

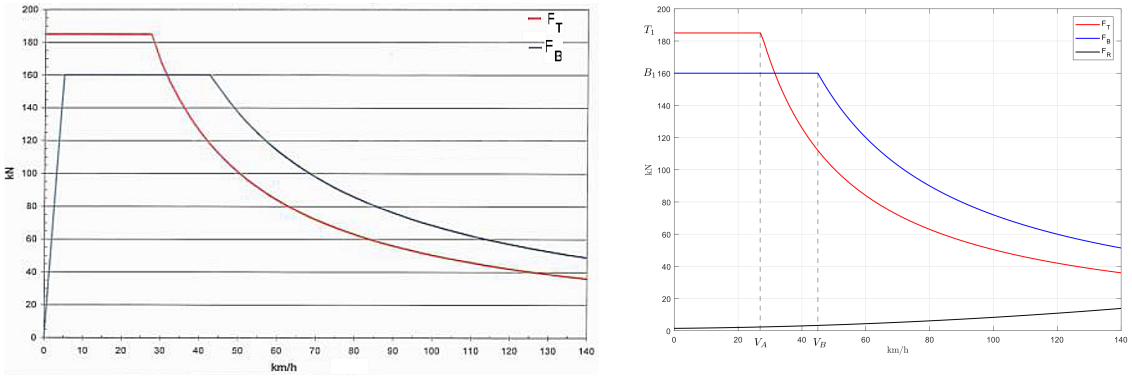
Concerning train traction, F_T, and braking forces, F_B, Figure 3a shows that they depend on train velocity. So, it will be needed a model that can fit those curves. Therefore, data from Figure 3a were approximated using the following equations, respectively, for traction and braking regimes:

$$F_T = \begin{cases} T_1, & \text{if} \quad v \leq V_a, \\ \dfrac{T_1 V_a}{v} = \dfrac{P_a}{v}, & \text{if} \quad v > V_a. \end{cases} \tag{4}$$

$$F_B = \begin{cases} B_1, & \text{if} \quad v \leq V_b, \\ \dfrac{B_1 V_b}{v} = \dfrac{P_b}{v}, & \text{if} \quad v > V_b. \end{cases} \tag{5}$$

Figure 3b presents obtained results for traction, braking and resistance curves of the modelled train. It can be seen that traction and braking forces have some similarities mainly due to train characteristics. For low velocities, constant forces are available in traction and braking modes, T_1 and B_1, respectively. In medium and high velocities, there is a reduction in traction force because train

enters in a constant power region. In this case, traction and braking characteristic are represented by available power, P_a and P_b, divided by train velocity.



(**a**) Real (**b**) Approximated

Figure 3. Train traction and braking forces vs. train velocity.

Diving the total applied force, F, by the mass, M, gives train acceleration, a in Equation (6):

$$\sum F = Ma \tag{6}$$

The effect of moments of inertia, due to train rotating parts, is considered, in the dynamic model, adding to Equation (6) a mass correction factor, γ, as shown in Equation (7), being M_e is the equivalent mass:

$$M_e = (1+\gamma)M \tag{7}$$

Usually γ takes values from 0.06 to 0.11 depending on the type of train and its traction system (centralized or distributed) [28].

Therefore the equation for train acceleration is:

$$a = \frac{dv}{dt} = \frac{dv}{dx}\frac{dx}{dt} = \frac{dv}{dx}v = \frac{\sum F}{(1+\gamma)M} = \frac{F_T - F_B - F_R - F_G}{(1+\gamma)M} \tag{8}$$

Starting with Equation (8) equations for time and distance, associated to each one of the four driving regimes, related to the train journey, are:

$$\Delta t = t_f - t_i = \int_{v_i}^{v_f} \frac{(1+\gamma)M}{F_T - F_B - F_R - F_G}\, dv \tag{9}$$

$$\Delta x = x_f - x_i = \int_{v_i}^{v_f} \frac{(1+\gamma)Mv}{F_T - F_B - F_R - F_G}\, dv \tag{10}$$

Considering the traction force and the velocity of the train, the mechanical power, P_{mec}, is determined from the following equation:

$$P_{mec}(t) = F_T(t) \times v(t) \tag{11}$$

As the purposed algorithm objective is the minimization of energy consumption, and considering the values of mechanical power Equation (11), the amount of energy, E_c, necessary to accomplish a journey is:

$$E_c = \int_0^T P_{mec}(t)\, dt \tag{12}$$

3.2. Train Motion Simulator

In order to evaluate total power and energy, associated to the OSP, and needed for SA algorithm a motion simulator is required. Developed TMS uses a state machine whose states are each one of the four train driving regimes (acceleration, cruising, coasting and braking). Transitions between states are defined based on initial values generated for the velocities at start of cruising and start of braking, respectively, v_{op} and v_{bk}. In each state, velocity limits, associated to the line, must be considered. The motion simulator runs with a sampling time of one second. In each time interval, from $t-1$ to t, train acceleration is considered constant with a value, a_t, obtained from Equation (8). In each of the four regimes, determination of train velocity and position, v_t and x_t, uses Equations (13) and (14).

$$v_t = a_t t + v_{t-1} \tag{13}$$

$$x_t = a_t \frac{t^2}{2} + v_t t + x_{t-1} \tag{14}$$

3.3. OSP Generation

The OSP generation mechanism starts with a random choice of values for cruising, v_{op}, and braking, v_{bk}, velocities. Each generated OSP uses the four driving regimes presented in the introduction.

Figure 4 shows the initial shape of the speed profile. Acceleration and braking phases uses maximum available force from Equations (4) and (5). Coasting phase is dependent of train's inertia. So, with the choice of v_{op} and v_{bk} the speed profile is completely characterized by times, t_1, $t_2 - t_1$, $t_3 - t_2$ and $t_4 - t_3$, associated with each driving regime. Calculation of these times assumes an ideal railway line without velocity constraints and slopes.

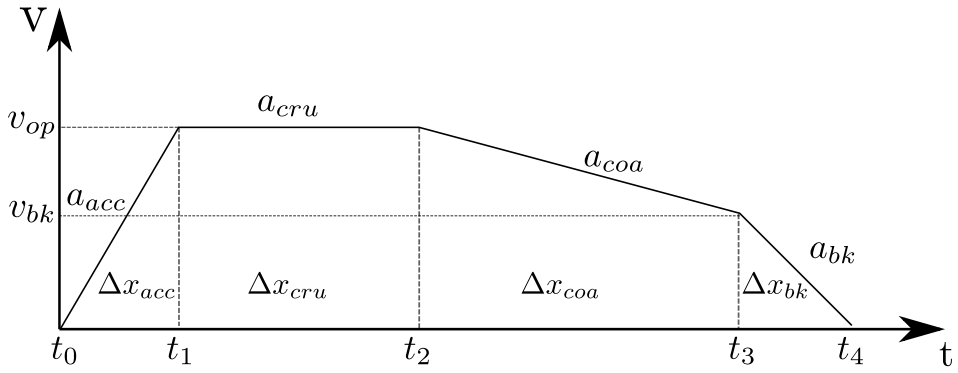


Figure 4. Initial shape of speed profile.

TMS then uses the obtained speed profile shape for updating, at each time interval, train acceleration, velocity and position, taking into account line slopes and velocity limits.

Output of OSP generation is a new speed profile, its respective energy, time and distance spent in the journey.

Figures 5 and 6 presents some OSP results, related to the modelled train, for lines with and without constraints. Associated energies, times and distances are in Tables 1 and 2. Motion simulator outputs are used by SA algorithm for identification of the best solution as shown in the following.

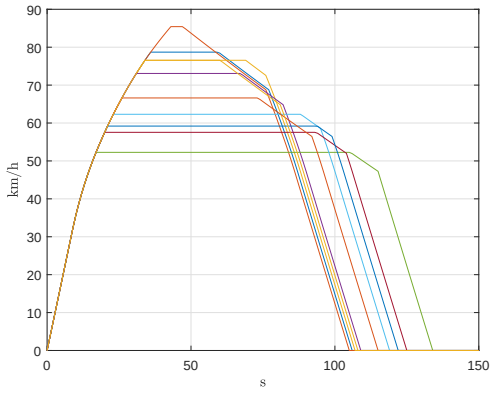


Figure 5. OSP generated without speed constraints.

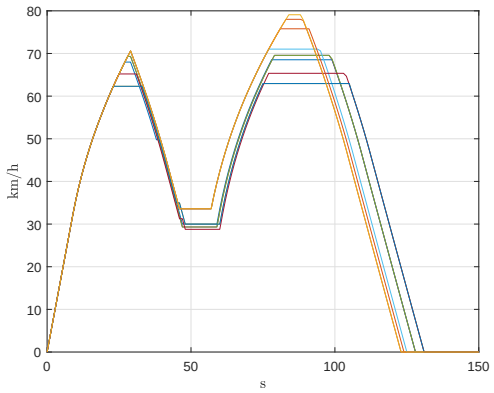


Figure 6. OSP generated with speed constraints.

Table 1. OSP energies, times and distances.

Energy (kWh)	Time (s)	Distance (m)
17.10	106	1710
18.16	105	1726
15.78	115	1722
13.03	129	1714
13.32	124	1711
11.43	133	1722
13.32	124	1711
14.94	121	1713
15.40	114	1720
15.77	114	1723

Table 2. OSP energies, times and distances.

Energy (kWh)	Time (s)	Distance (m)
21.68	125	1724
19.36	132	1718
18.70	134	1713
21.12	126	1726
21.51	125	1718
18.93	132	1711
19.80	132	1726
20.63	129	1714
21.98	125	1727
21.12	126	1726

4. Speed Profile Optimization

Having developed a model able to generate speed profiles and its associated energy, time and space, subjected to velocity limits, scheduling and line gradients, the problem is now how to find the one that optimizes consumed energy, E_c, that is:

Minimize:

$$E_c = \sum_{t=1}^{N} e_t \tag{15}$$

Constrained to:

$$v_t < v_{max} \quad \text{for} \quad 1 \leq t \leq J \tag{16}$$

$$t_{total} < t_{max} \quad \text{for} \quad 1 \leq t \leq J \tag{17}$$

$$a_t < min\left[a_{max}, \frac{FT - FB - FR - FG}{(1+\gamma)M}\right] \quad \text{for} \quad 1 \leq t \leq J \tag{18}$$

With the number of time steps considered in the journey between two stations being J and e_t the consumed energy in each time step, v_t, and a_t are, respectively, velocity and acceleration at any time t. t_{total} and t_{max} are total and maximum journey times. v_{max}, and a_{max} are, respectively, train maximum velocity and acceleration.

Also the solution should be able for real time implementation so that it can be used as a connected DAS.

4.1. Simulated Annealing

The proposed optimization problem is a large combinatorial one, associated to decisions about the time and velocities, for each of the four considered traction regimes, constrained to line slopes and velocity limits. So, a meta-heuristic approach using SA is tried. SA algorithm intends to mimic the metallurgical process of obtaining perfect crystalline structures, in a solid material, by first heating it up to high temperatures and then letting it gradually cool down. The algorithm has two major steps associated to the processes of cooling and getting new inputs. Thus, it needs definition of the cooling scheme and a method that controls how to vary its inputs. Besides, it needs an initial temperature definition, an objective function, associated to the minimization problem, and a termination criteria. Figure 7 shows a generic flow chart of SA algorithm.

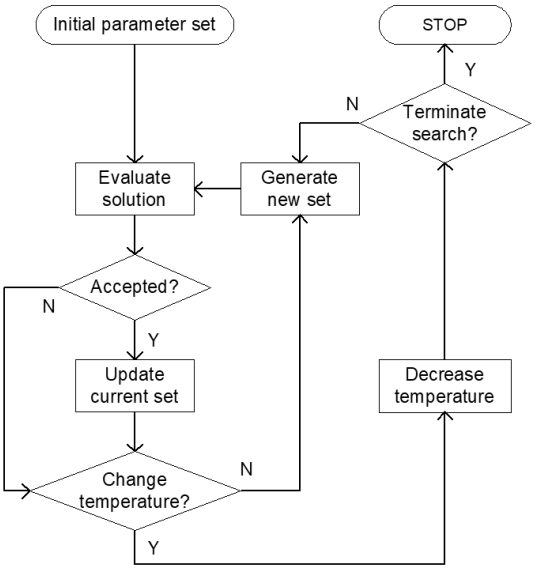


Figure 7. Simulated Annealing flow chart [29].

After parameter initialization SA algorithm decides if the new problem solution is accepted based on actual value of its objective function, O_{i+1}. If O_{i+1} is smaller than the last one, O_i, the new solution is accepted and stored. If not, actual solution can still be admitted if the value given by the Boltzmann distribution:

$$\exp - \left[\frac{O_{i+1} - O_i}{T_{i+1}} \right] \tag{19}$$

is greater than a random number between $[0, 1]$. This is the main feature of SA that enables it to escape from local minimum. In this work, the objective function, to minimize, is made as a combination of journey energy, E_c, and absolute difference between journey and schedule times, $|t_{total} - t_{max}|$, considering a penalty factor λ:

$$O = E_c + \lambda \, |t_{total} - t_{max}| \tag{20}$$

Next step of the algorithm re-anneals (SA algorithm can re-anneal, a certain number of times, with the same temperature. In these work re-annealing is not used). Algorithm proceeds with temperature decrease using:

$$T_{i+1} = s \times T_i \tag{21}$$

For the cooling rate, s, it uses 0.8. Initial value of temperature uses [22]:

$$T = - \left[\frac{\Delta O}{\ln(0.8)} \right] \tag{22}$$

If termination criteria is satisfied algorithm stops (this work uses 100 iterations as the termination criteria). If not, a new solution is tried.

4.2. Optimization Algorithm

Having presented the developed state machine, associated to TMS, as well as the OSP generation mechanism, SA algorithm principles and the problem objectives and constraints, Figure 8 shows the optimization algorithm flowchart implemented for solution of our problem.

After initialization, optimization procedure is as follows:

1. Use Equations (23) and (24) to generate values of v_{op} and v_{bk}.

$$v_{op}^{i+1} = v_{op}^{i} + \Delta v_{op}^{i+1} \tag{23}$$

$$v_{bk}^{i+1} = v_{bk}^{i} + \Delta v_{bk}^{i+1} \tag{24}$$

   where Δv_{op}^{i+1} and Δv_{bk}^{i+1} are values randomly picked from a vector containing velocities from a minimum, v_{min}, to a maximum v_{max}.
2. Use TMS and OSP generation mechanism to output a new speed profile and its associated energy and journey time.
3. If the errors in travelling time and space are admissible then use obtained speed profile energy and time for SA algorithm. If not go to rstep 1 and generate new values of v_{op} and v_{bk}.
4. If at maximum number of iterations the algorithm stops and gives the optimized solution. If not go to step 1 and generate new values of v_{op} and v_{bk}.

Note that there are two different processes for v_{op} and v_{bk} generation, controlled by a flag:

If, in the last iteration, SA algorithm accepted the solution Δv_{op}^{i+1} is randomly picked from a vector with $v_{min} = -10$ and $v_{max} = 10$. Δv_{bk}^{i+1} is randomly chosen from a vector with $v_{min} = -10$ and $v_{max} = v_{op}^{i+1}$.

If not, Δv_{op}^{i+1} is randomly selected from a vector with $v_{min} = 0$ and v_{max} equal to maximum train velocity. Δv_{bk}^{i+1} is randomly picked from a vector with $v_{min} = 0$ and $v_{max} = v_{op}^{i+1}$.

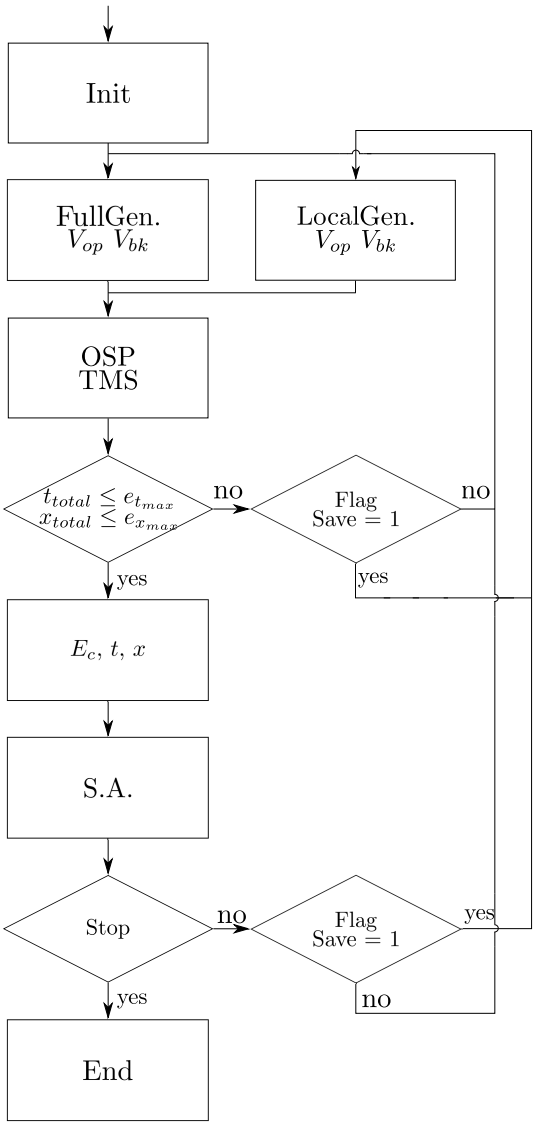


Figure 8. Optimization Algorithm Flowchart.

5. Results

In order to prove its effectiveness the algorithm was tested in four different line sections, between eight railway stations. Table 3 presents, for tested lines, the schedule for journey time, total travelling distance between stations, and spent energy without the use of the algorithm. All measurements have been taken during regular service with a sampling time of 2 seconds. Distance is incremented in 20 m steps. Table 4 shows speed limits. Inputs associated to TMS, OSP and SA are in Table 5. Figure 9 shows slope data.

Table 3. Cases Studied.

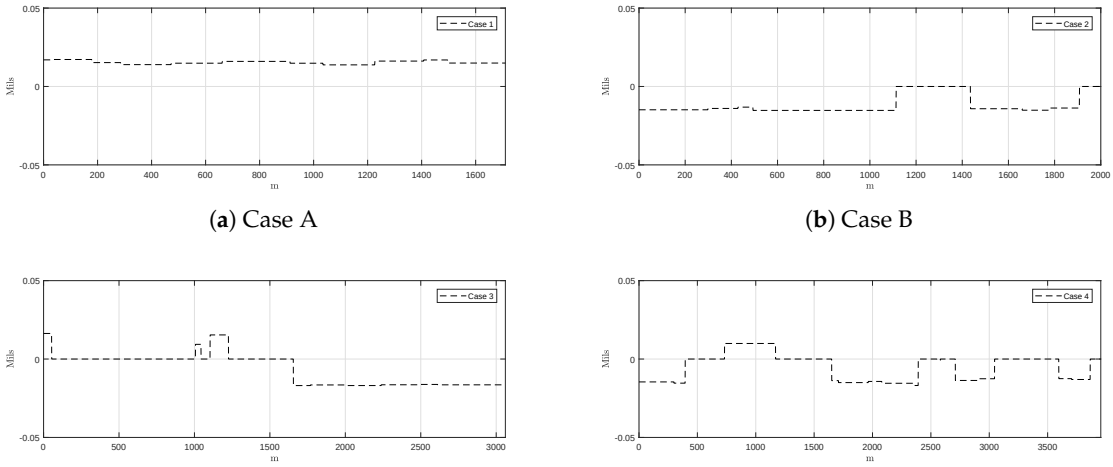
Case	Distance (m)	Time (s)	Energy (kWh)
A	1710	120	14.78
B	2000	120	11.44
C	3061	210	10.83
D	3955	200	15.10

Table 4. Speed limits.

Case	Start (m)	End (m)	Speed Limit (km/h)
A	0	1710	110
B	0	2000	120
C	0	440.9	45
	440.9	3060	100
D	0	3727	110
	3727	3955	80

Table 5. TMS, OSP and SA Data.

Train and SA data					
$F_{T_{max}}$	185	kN	a_{max}	1	m/s
$F_{B_{max}}$	160	kN	γ	0.05	
M	117	ton	it_{max}	100	
v_{max}	140	km/h	s	0.8	



(**a**) Case A (**b**) Case B

(**c**) Case C (**d**) Case D

Figure 9. Line Slopes.

The algorithm was implemented in Matlab, using a computer equipped with an Intel Core i7 at 2.10 GHz and 8 Gb of RAM. The computer is running a 64-bit version of Windows 10 Pro. Algorithm results, for energy consumption and generated OSP, are compared with data acquired from the studied railways lines.

For the first case (case A), total distance is 1710 m and allowed time is 120 s, as can be seen in Table 3. The train was operating without any DAS, so, driver was only concerned with schedule accomplishment without violating velocity limits. Current applied speed profile is based on driver's experience and is presented in Figure 10 (dashed black line). In what concerns velocity limits both velocities profiles are always below velocity limits. Acquired data shows, for this journey, a total traction energy of 14.78 kWh.

Taking into account line profile and respective scheduling for the trip, the algorithm simulation results estimate an energy consumption of 12.61 kWh for the journey, so, an energy consumption reduction up to 2 kWh. Figure 10 (solid blue line) shows optimum speed profile given by the algorithm.

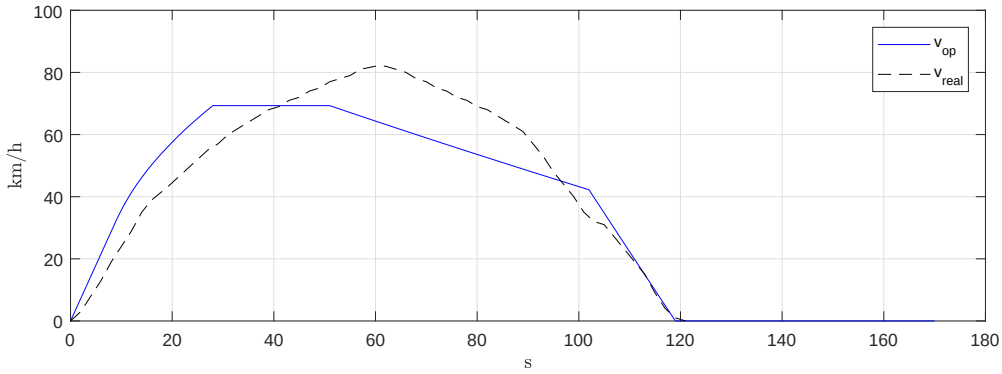


Figure 10. Case A—Actual (dashed black) and proposed (solid blue) speed profiles.

Figure 11 presents a graphic bar where algorithm results and real measurements are compared in terms of energy consumption, time and distance error.

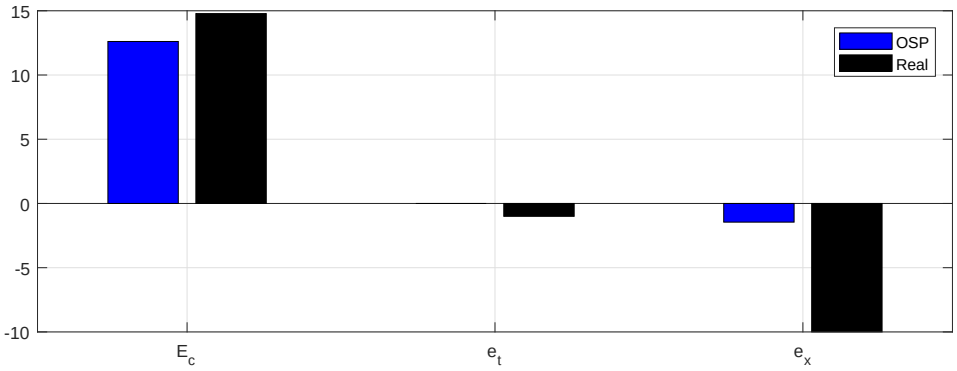


Figure 11. Case A—Real (black) and simulated (blue) energies (kWh), traveling times (s) and distance errors (m).

It can be seen that algorithm OSP output uses maximum time available for the trip (actual measurement, taking into account train logger sampling rate, shows that the driver also arrived in schedule). The algorithm output advices the driver to achieve cruising velocity faster in order to decrease top velocity needed to accomplish the total line length. A lower value of v_{op} enables a consumed energy decrease due to a reduction of motion resistance forces. The algorithm needed 7 s to output the solution, so it enables its use in real-time applications as intended. Distance errors are negligible considering the train logger step increments (20 m).

For the second studied case (case B) a total travelling distance of 2000 m to be accomplished in 120 s was considered. Train's operation is the same as previous case. Figure 12 (black dashed line) presents velocity measurements. Energy consumption is 11.44 kWh.

Running the algorithm outputs an OSP with an energy consumption of, approximately, 10.06 kWh. Figure 12 shows obtained algorithm speed profile (solid blue line). Once more, it can be seen that the same logic is used. OSP output advices the driver to accelerate the train with the maximum allowed acceleration up to cruising velocity. Achieving a lower cruising velocity, less energy is spent in overcoming tractive effort resistance in order to maintain a constant acceleration.

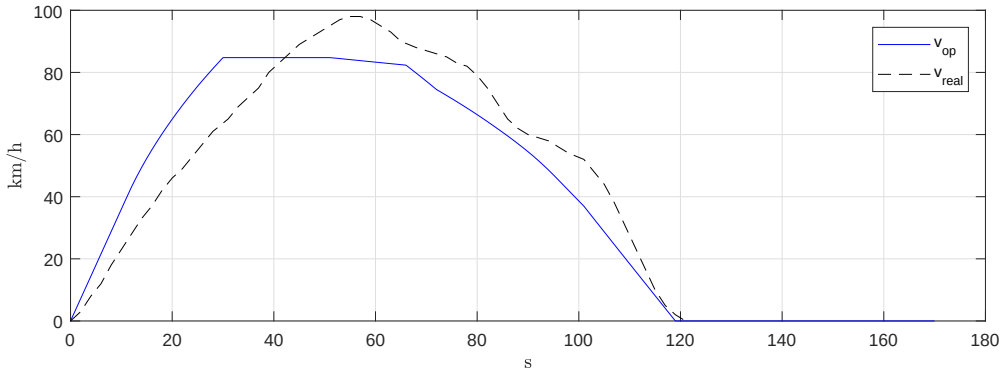


Figure 12. Case B—Actual (dashed black) and proposed (solid blue) speed profiles.

Figure 13 shows energy consumption, time and distance error for the OSP and real measurements. Again, reduction of v_{op} velocity, allied to the use of all travelling time, enables to achieve an energy consumption reduction of, in this case, 1 kWh. Algorithm OSP output uses again maximum time available for the trip. Train logger output shows that the driver also arrived in schedule. Distance errors are within the 20 m train logger step increment. Algorithm running time was 9 s.

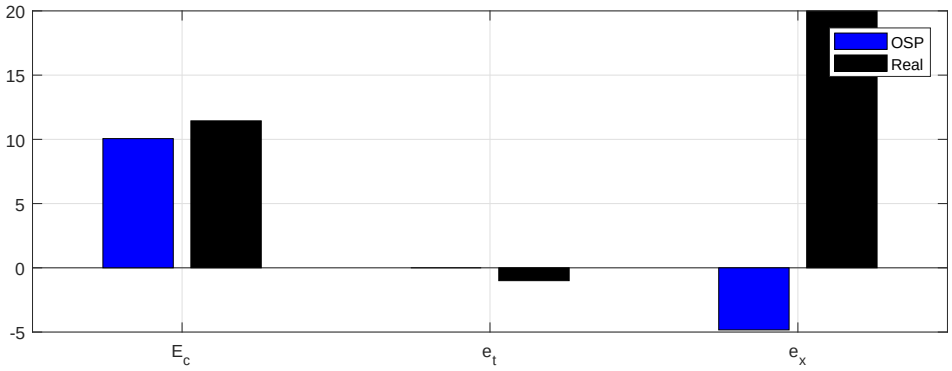


Figure 13. Case B—Real (black) and simulated (blue) energies (kWh), traveling times (s) and distance errors (m).

The third considered case (case C), as seen in Table 4, has a maximum allowed velocity of 45 km/h during, approximately, the first 440 m. After this distance speed limit increases to 100 km/h. Figure 14 shows actual speed profile (black dashed line).

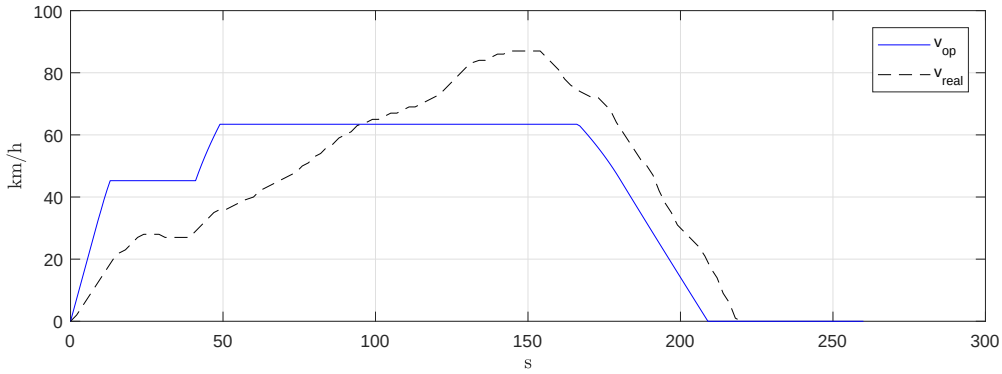


Figure 14. Case C—Actual (dashed black) and proposed (solid blue) speed profiles.

Algorithm OSP output (also in Figure 14, as the solid blue line) shows a total energy consumption of 9.47 kW, enabling a reduction of 1 kW and a speed profile that meets scheduling. Also, the algorithm satisfies imposed velocity constraints, at the beginning of the journey, and as a running time of 15 s so enabling real-time operation as desired. Numerical results of OSP and real measurements can be seen in Figure 15.

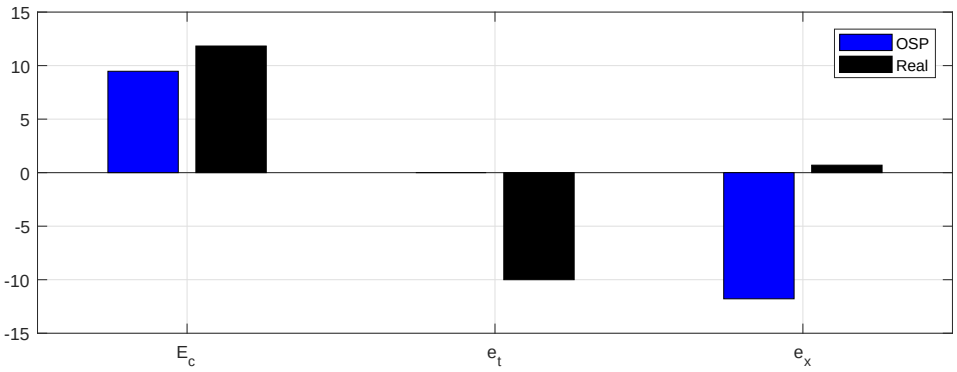


Figure 15. Case C—Real (black) and simulated (blue) energies (kWh), traveling times (s) and distance errors (m).

In this case distance between stations is 3061 m and desired travelling time 210 s. Simulated results show no error in time and about 0.5% in distance. Logged data shows a 10 delay in time. In this case the error in distance is below the step size so it can be considered null.

For the last case (case D) a total distance of 3955 m to be accomplished in 200 s is considered. As shown in Table 4, at a travelling distance of 3727 m, velocity limit changes from 110 km/h to 80 km/h. Nevertheless, despite this being used by OSP algorithm, actual velocity, at this distance, is always lower so this constraint does not appear in the simulated output profile. Figure 16 shows both velocity profiles (in dashed black the measured train velocity and in blue OSP output). Besides the energy consumption reduction, OSP output advices a speed profile that accomplishes the original scheduling imposed by railway operator. The improvements to energy consumption can be seen in Figure 17. Simulated results have no error in travelling time. Real one has a 14 s delay. Distance errors are negligible in both cases. Again, the algorithm running time is compatible with real-time DAS.

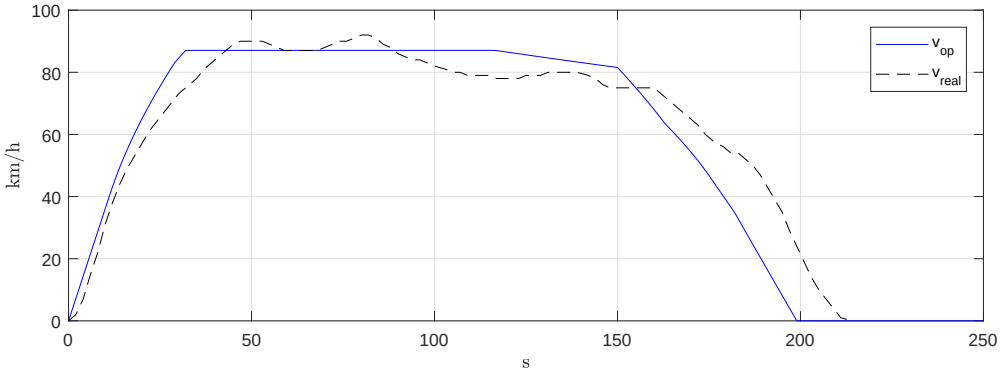


Figure 16. Case D—Actual (dashed black) and proposed (solid blue) speed profiles.

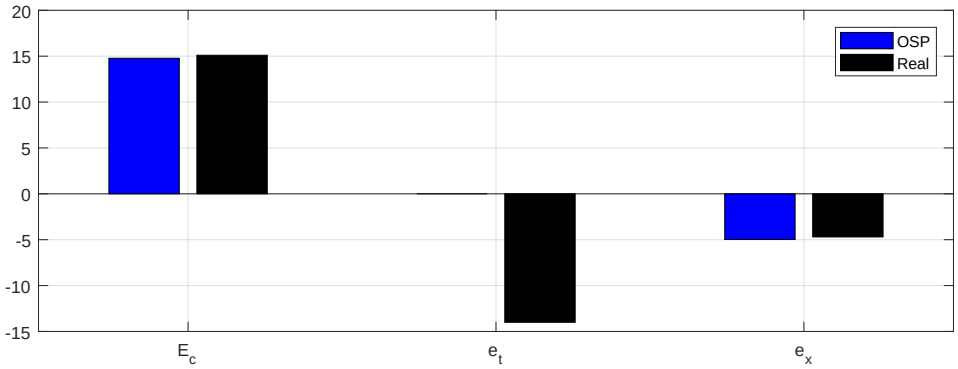


Figure 17. Case D—Real (black) and simulated (blue) energies (kWh), traveling times (s) and distance errors (m).

Table 6 shows, besides time and distance results, actual consumed energies, energies outputs using the proposed speed profiles and algorithm running times. Therefore, for all studied cases, results show a lower energy consumption allied to the fulfilment of maximum allowed times. For an easier evaluation, algorithm output is presented besides actual data measurements. Note that all simulated cases uses train mass (117 ton) and 300 passengers (2.1 ton) for train total weight.

Table 6. Algorithm Results.

Stations	Distance (m)		Time (s)		Energy (kWh)		Weight (ton)		Running Time (s)
	Real	OSP	Real	OSP	Real	OSP	Real	OSP	
Case A	1720	1709	121	120	14.78	12.61	118	143.8	7
Case B	1980	2008	121	120	11.44	10.06	121	143.8	9
Case C	3060	3073	220	210	10.83	9.47	117	143.8	15
Case D	3960	3960	214	200	15.10	14.78	119	143.8	20

6. Conclusions and Future Work

This paper presented a new algorithm capable of finding train OSP, between stations, in order to achieve minimum energy consumption. Constraints associated with the line, such as its gradients and velocity limits, are considered. The proposed algorithm was tested against current train operation, in four-line sections associated with four pairs of consecutive stations. Obtained results show that its use will enable considerable energy savings. Also, its running time is compatible with real-time operation, so it is capable of application as a connected DAS as proposed.

Future work intends to start its implementation as C code, associated with GPS data, for use in an Android-based tablet, as a DAS. Also, the use of regenerative braking in developed TMS will be considered.

Author Contributions: For this paper investigation as well as the methodology, software and validation was conducted by A.R. under supervision of A.A., A.C. and J.S. The original draft preparation was made by A.R. and A.A. All authors contributed to the main revision and editing.

Funding: This research was funded by FCT (Fundação para a Ciência e a Tecnologia) under grant PD/BD/114104/2015.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

ACO	Ants Colony Optimization
ANN	Artificial Neural Network
ATO	Automatic Train Operation
DAS	Driver Advisory System
DP	Dynamic Programming
EA	Evolutionary Algorithm
GA	Genetic Algorithm
IBEA	Indicator—Based Evolutionary Algorithm
MARK	Minimum-Allele-Reserve-Keeper
MMAS	MAX-MIN Ant System
OSP	Optimal Speed Profile
PMP	Pontryagyn's Maximum Principle
PSO	Particle Swarm Optimization
SA	Simulated Annealing
TMS	Train Motion Simulator

References

1. Fernández-Rodríguez, A.; Fernández-Cardador, A.; Cucala, A.P. Energy efficiency in high speed railway traffic operation: A real-time ecodriving algorithm. In Proceedings of the 2015 IEEE 15th International Conference on Environment and Electrical Engineering (EEEIC), Rome, Italy, 10–13 June 2015; pp. 325–330.
2. Scheepmaker, G.M.; Goverde, R.M.; Kroon, L.G. Review of energy-efficient train control and timetabling. *Eur. J. Oper. Res.* **2017**, *257*, 355–376. [CrossRef]
3. Howlett, P.; Milroy, I.; Pudney, P. Energy-efficient train control. *Control Eng. Pract.* **1994**, *2*, 193–200. [CrossRef]
4. Qu, J.; Feng, X.; Wang, Q. Real-time trajectory planning for rail transit train considering regenerative energy. In Proceedings of the 17th International IEEE Conference on Intelligent Transportation Systems (ITSC), Qingdao, China, 8–11 October 2014; pp. 2738–2742.
5. Milroy, I.P. Aspects of Automatic Train Control. Ph.D. Thesis, Loughborough University, Leicestershire, UK, 1980.
6. Liu, R.R.; Golovitcher, I.M. Energy-efficient operation of rail vehicles. *Transp. Res. Part A* **2003**, *37*, 917–932. [CrossRef]
7. Chang, C.S.; Sim, S.S. Optimising train movements through coast control using genetic algorithms. *IEE Proc. Electr. Power Appl.* **1997**, *144*, 65–73. [CrossRef]
8. Wong, K.K.; Ho, T.K. Dynamic coast control of train movement with genetic algorithm. *Int. J. Syst. Sci.* **2004**, *35*, 835–846. [CrossRef]
9. Acikbas, S.; Soylemez, M.T. Coasting point optimisation for mass rail transit lines using artificial neural networks and genetic algorithms. *IET Electr. Power Appl.* **2008**, *2*, 172–182. [CrossRef]
10. Howlett, P.G.; Pudney, P.J. Practical Strategy Optimisation. In *Energy-Efficient Train Control*; Springer: London, UK, 1995; pp. 285–297.
11. Su, S.; Tang, T.; Chen, L.; Liu, B. Energy-efficient train control in urban rail transit systems. *Proc. Inst. Mech. Eng. Part F* **2015**, *229*, 446–454. [CrossRef]
12. Huang, Y.; Ma, X.; Su, S.; Tang, T. Optimization of Train Operation in Multiple Interstations with Multi-Population Genetic Algorithm. *Energies* **2015**, *8*, 14311–14329. [CrossRef]
13. Chevrier, R.; Marliere, G.; Vulturescu, B.; Rodriguez, J. *Multi-Objective Evolutionary Algorithm for Speed Tuning Optimization with Energy Saving in Railway: Application and Case Study*; RailRome: Rome, Italy, 2011.
14. Lu, S.; Hillmansen, S.; Ho, T.K.; Roberts, C. Single-Train Trajectory Optimization. *IEEE Trans. Intell. Transp. Syst.* **2013**, *14*, 743–750. [CrossRef]
15. Huang, Y.; Yang, C.; Gong, S. Energy Optimization for Train Operation Based on an Improved Ant Colony Optimization Methodology. *Energies* **2016**, *9*, 626. [CrossRef]

16. Domínguez, M.; Fernández-Cardador, A.; Cucala, A.P.; Gonsalves, T.; Fernández, A. Multi objective particle swarm optimization algorithm for the design of efficient ATO speed profiles in metro lines. *Eng. Appl. Artif. Intell.* **2014**, *29*, 43–53. [CrossRef]
17. Kim, K.; Chien, S.I.J. Optimal Train Operation for Minimum Energy Consumption Considering Track Alignment, Speed Limit, and Schedule Adherence. *J. Transp. Eng.* **2011**, *137*, 665–674. [CrossRef]
18. Xie, T.; Wang, S.; Zhao, X.; Zhang, Q. Optimization of Train Energy-Efficient Operation Using Simulated Annealing Algorithm. In *Intelligent Computing for Sustainable Energy and Environment*; Li, K., Li, S., Li, D., Niu, Q., Eds.; Springer: Berlin/Heidelberg, Germany, 2013; pp. 351–359.
19. Busetti, F. Simulated Annealing Overview. 2003. Available online: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.66.5018&rep=rep1&type=pdf (accessed on 27 September 2018).
20. Chevrier, R.; Pellegrini, P.; Rodriguez, J. Energy saving in railway timetabling: A bi-objective evolutionary approach for computing alternative running times. *Transp. Res. Part C* **2013**, *37*, 20–41. [CrossRef]
21. Keskin, K.; Karamancioglu, A. Energy-Efficient Train Operation Using Nature-Inspired Algorithms. *J. Adv. Transp.* **2017**, *2017*, 6173795. [CrossRef]
22. Ben-Ameur, W. Computing the Initial Temperature of Simulated Annealing. *Comput. Optim. Appl.* **2004**, *29*, 369–385. [CrossRef]
23. Howlett, P.G.; Pudney, P.J. *Energy-Efficient Train Control*; Springer: London, UK, 1995.
24. Douglas, H.; Weston, P.; Kirkwood, D.; Hillmansen, S.; Roberts, C. Method for validating the train motion equations used for passenger rail vehicle simulation. *Proc. Inst. Mech. Eng. Part F* **2017**, *231*, 455–469. [CrossRef]
25. Su, S.; Tang, T.; Li, X. Driving strategy optimization for trains in subway systems. *Proc. Inst. Mech. Eng. Part F* **2016**, *232*, 369–383. [CrossRef]
26. Davis, W.J. Tractive Resistance of Electric Locomotives and Cars. *Gen. Electr. Rev.* **1926**, *29*, 685–708.
27. Rochard, B.P.; Schmid, F. A review of methods to measure and calculate train resistances. *Proc. Inst. Mech. Eng. Part F* **2000**, *214*, 185–199. [CrossRef]
28. Yi, S. *Principles of Railway Location and Design*; Academic Press: Cambridge, MA, USA, 2018; pp. 133–135.
29. Chibante, R.; Araujo, A.; Carvalho, A. *Simulated Annealing, Theory with Applications*; IntechOpen: London, UK, 2010.